



ANDROID



PROGRAMIRANJE KORISNIČKIH
INTERFEJSA 2025/2026

ELEKTOTEHNIČKI FAKULTET
UNIVERZITETA U BEOGRADU

Operativni sistem za mobilne telefone i tablete

Prva verzija – septembar 2008

Google

Stotine miliona telefona koriste Android u preko 190 zemalja

Veliki broj aplikacija za Android koje se mogu preuzeti sa Google Play-a

Šta je Android?

Verzije Androida



API nivo

Svaka verzija Androida podržava određeni API (dostupne funkcionalnosti)

Npr:

- Android 10 – api level 29
- Android 11 – api level 30
- Android 12 – api level 31, 32
- Android 13 – api level 33
- Android 14 – api level 34
- Android 15 – api level 35
- Android 16 – api level 36

Instalacija alata

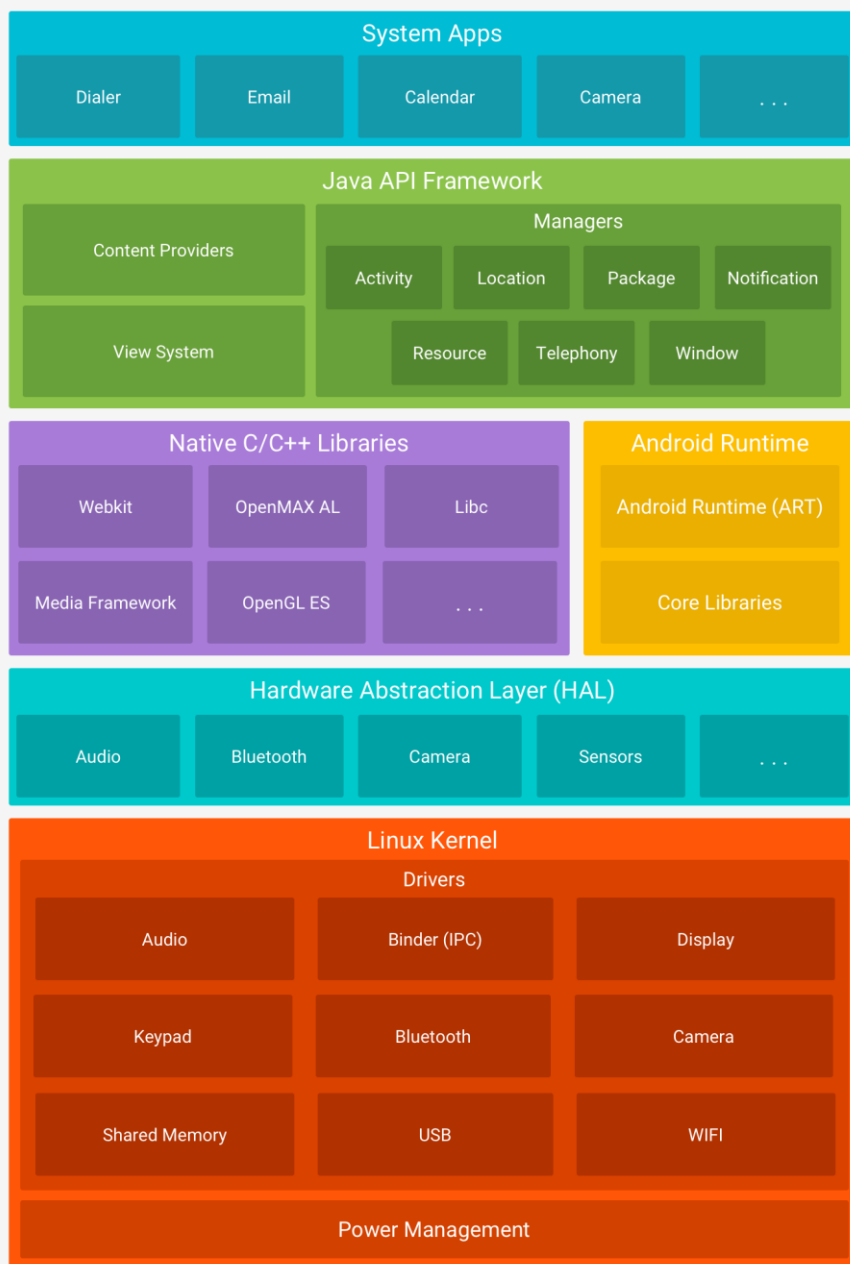
Android Studio

<https://developer.android.com/studio/>

Dokumentacija

<https://developer.android.com/guide/>

<https://developer.android.com/reference/android/package-summary>



Arhitektura Androida

Arhitektura Androida

Linux kernel – osnova Android platforme

Apstrakcija hardvera – ovaj sloj sadrži biblioteke koje standardizuju pristup hardverskim komponentama, i svaki hardverski modul ima svoj interfejs za povezivanje hardvera sa višim slojevima sistema

Svaka Android aplikacija ima svoj proces koji koristi svoju instancu **Android Runtime-a**. ART (ranije Dalvik VM) koristi napredne tehnike kompajliranja tako da poboljšava preformanse aplikacije i upravljanja memorijom

Osnovni delovi Android sistema, kao što su ART i HAL, napisani su u **C/C++**. Android pruža Java API-je koji omogućavaju pristup funkcionalnostima ovih nativnih biblioteka.

Sve funkcionalnosti Android operativnog sistema dostupne su kroz **Java API-je**

- Activity Manager – kontroliše životni ciklus aplikacije.
- Content Providers – dozvoljava deljenje sadržaja sa drugim aplikacijama.
- Resource Manager – omogućava pristup resursima.
- Notifications Manager – obezbeđuje rad sa notifikacijama.
- View System – služi za kreiranje korisničkog interfejsa.

Poslednji sloj čine **sistemske aplikacije**, poput aplikacija za SMS, kalendar, kontakte i slično.

Kotlin

Multiplatformski programski jezik, kompatibilan sa Javom

<https://play.kotlinlang.org/>

```
fun main() {  
    var msgFromFunction = happyBirthday("Petar")  
    msgFromFunction += " Yey!"  
    print(msgFromFunction)  
}  
  
fun happyBirthday(name: String): String{  
    val msg: String = "Happy birthday, $name!"  
    /*  
     Comments  
    */  
    return msg  
}
```

```
fun main() {  
    val trafficLightColor = "Amber"  
  
    when (trafficLightColor) {  
        "Red" -> println("Stop")  
        "Yellow", "Amber" -> println("Slow")  
        "Green" -> println("Go")  
        else -> println("Invalid traffic-light color")  
    }  
}
```

```
fun main() {  
    var favoriteActor: String? = "Sandra Oh"  
    favoriteActor = null  
}
```



```
class SmartDevice(val name: String, val category: String) {  
    var deviceStatus = "online"  
  
    constructor(name: String, category: String, statusCode: Int) : this(name, category)  
        deviceStatus = when (statusCode) {  
            0 -> "offline"  
            1 -> "online"  
            else -> "unknown"  
        }  
    }  
    ...  
}  
  
var speakerVolume = 2  
get() = field  
set(value) {  
    field = value  
}
```

```
open class SmartDevice(val name: String, val category: String) {  
    ...  
}
```

```
class SmartTvDevice(deviceName: String, deviceCategory: String) :  
    SmartDevice(name = deviceName, category = deviceCategory) {  
}
```

Lambda izrazi

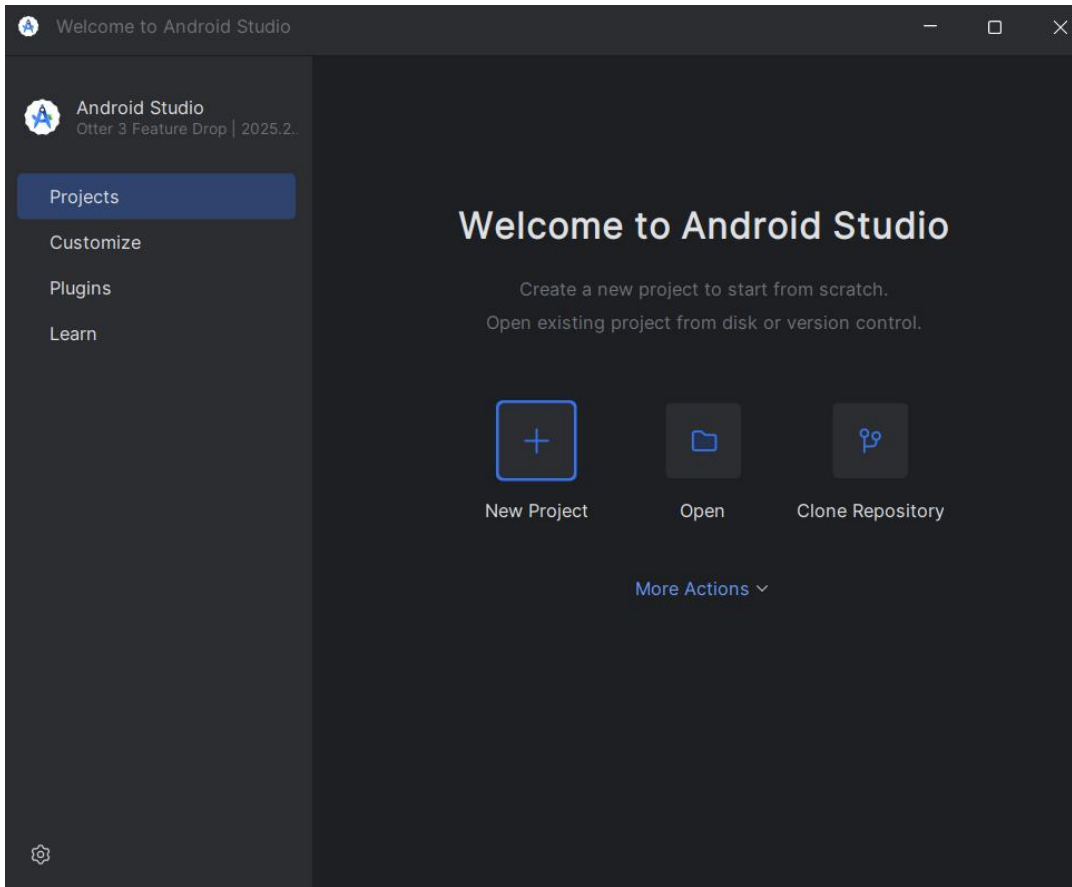
```
fun trickOrTreat(isTrick: Boolean, extraTreat: (Int) -> String): () -> Unit {  
    if (isTrick) {  
        return trick  
    } else {  
        println(extraTreat(5))  
        return treat  
    }  
}
```

```
val trick = {  
    println("No treats!")  
}  
  
val treat = {  
    println("Have a treat!")  
}
```

```
val coins: (Int) -> String = { quantity ->  
    "$quantity quarters"  
}
```

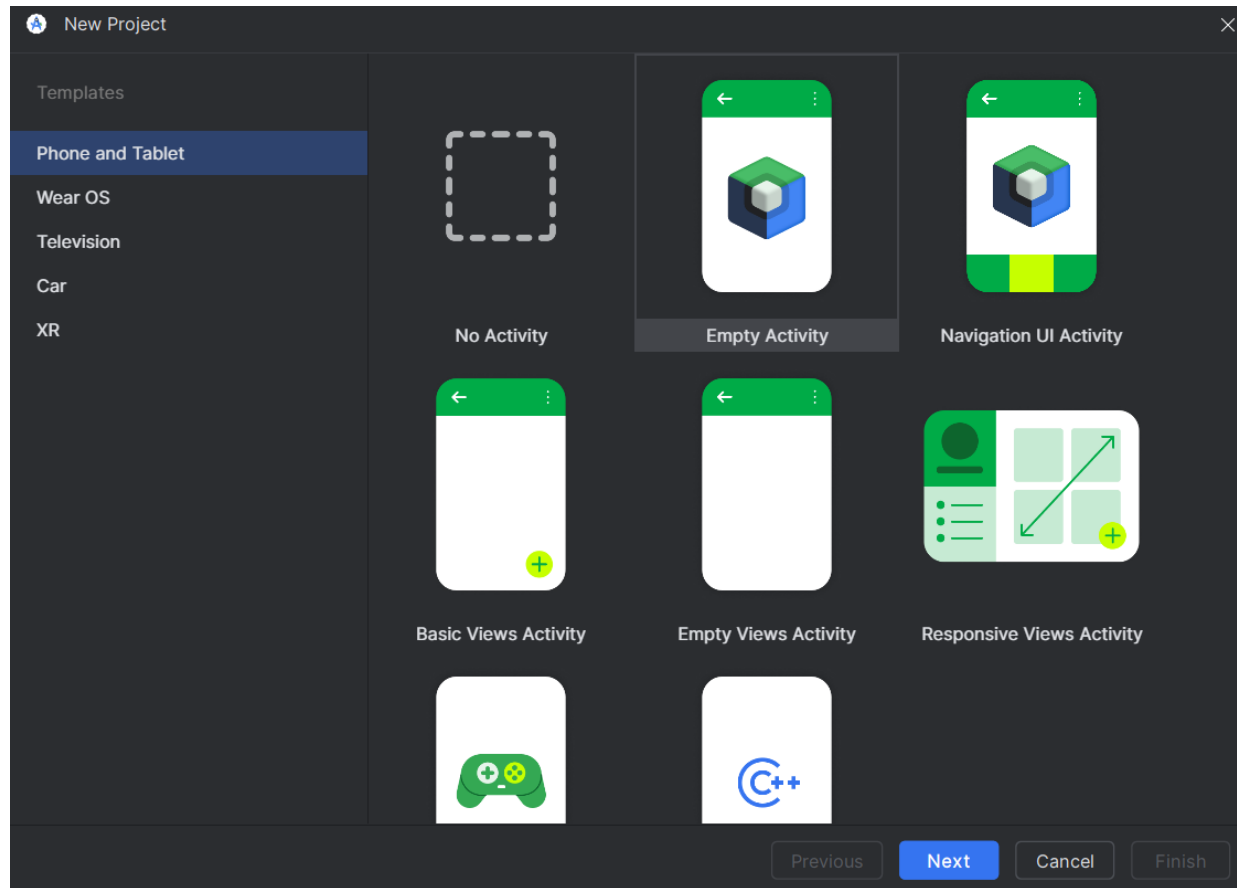


```
val coins: (Int) -> String = {  
    "$it quarters"  
}
```



Hello World

Hello World



Hello world

New Project

Empty Activity

Create a new empty activity with Jetpack Compose

Name

Package name

Save location

Minimum SDK

i Your app will run on approximately 99.2% of devices.
[Help me choose](#)

Build configuration language

Previous Next Cancel Finish

Struktura Android aplikacije

Manifest folder

- AndroidManifest.xml - Fajl koji opisuje fundamentalne karakteristike naše aplikacije

Java folder (Kotlin + java)

- Sadrži .kt fajlove našeg projekta kao i glavnu aktivnost koja se pokreće pri startovanju aplikacije

Res folder

- Drawable - Resursi koje možemo iscrtati na ekranu (npr. slike)
- Mipmap – Launcher ikonica aplikacije
- Values - XML fajlovi koji sadrže kolekcije stringova, boja...

Gradle skripte

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="My Application"
            android:theme="@style/Theme.MyApplication">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

```

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(paddingValues = innerPadding)
                    )
                }
            }
        }
    }
}

```

```

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

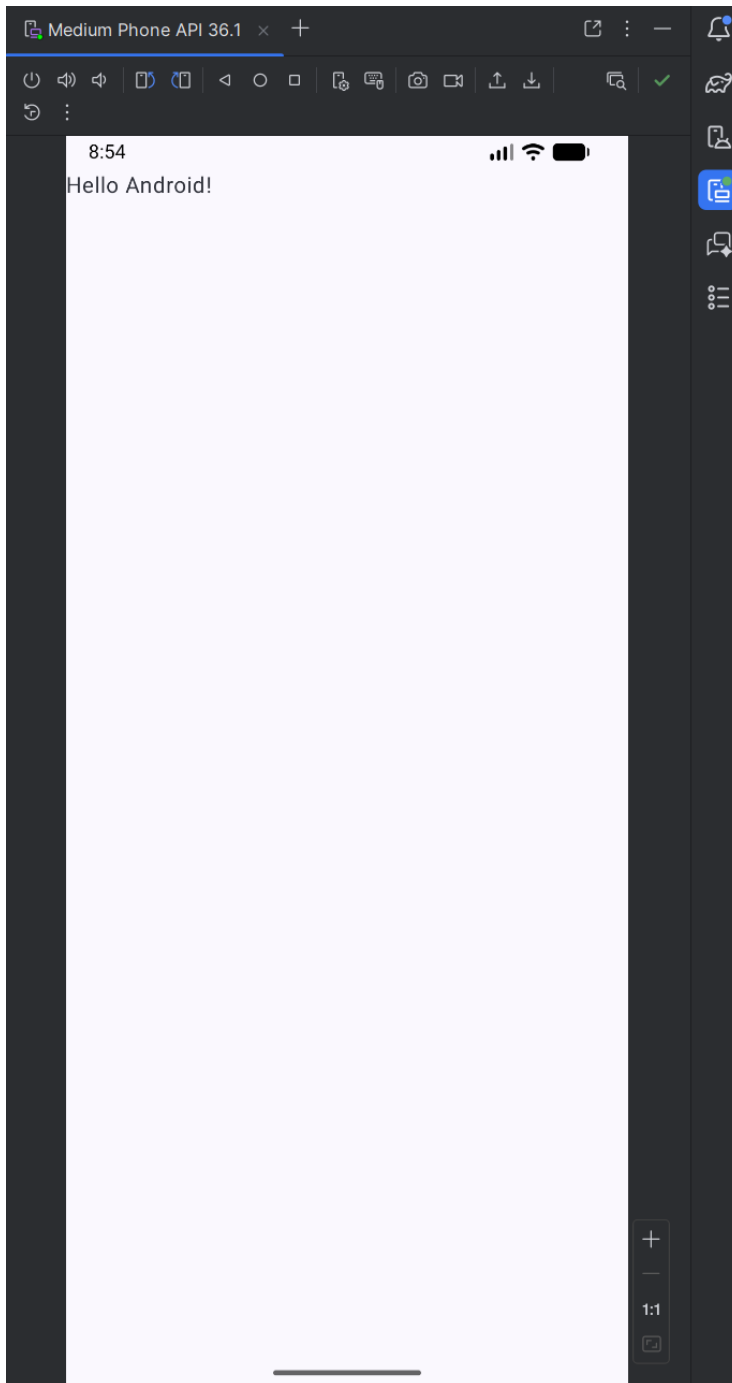
```

```

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    MyApplicationTheme {
        Greeting(name = "Android")
    }
}

```

MainActivity.kt



Pokretanje aplikacije

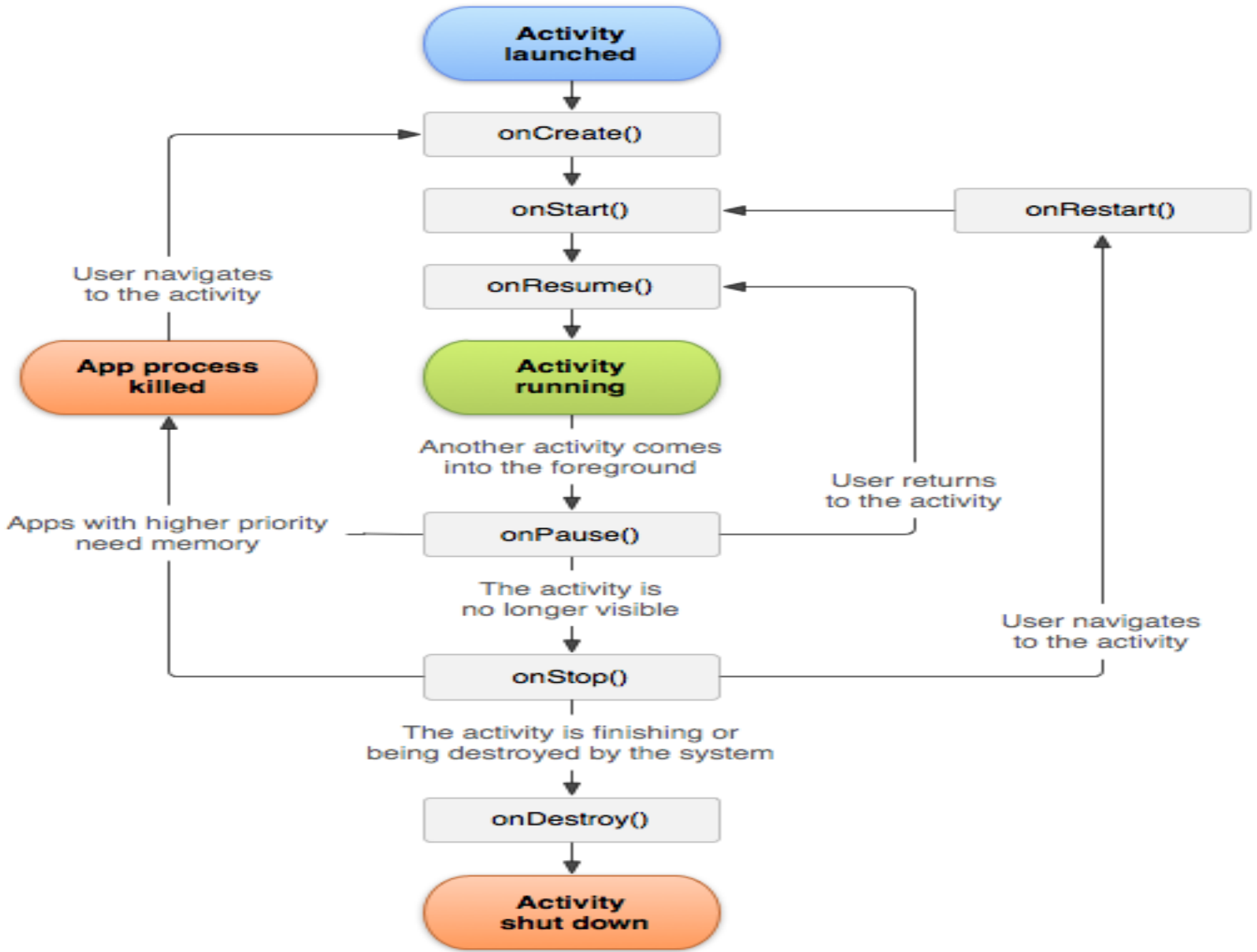
- Device Manager
- Android Emulator



Aktivnosti

Android sistem inicira program aktivnošću koja zove svoju onCreate() metodu

Svaka aktivnost ima svoj životni ciklus koji se sastoji od sekvence metoda koje startuju aktivnost i sekvence metoda koje gase aktivnost



Sr.No	Callback & Description
1	onCreate() Zove se nakon što je aktivnost prvi put kreirana
2	onStart() Zove se nakon što je aktivnost postala vidljiva korisniku
3	onResume() Zove se nakon što je aktivnost postala spremna za interakciju sa korisnikom
4	onPause() Zove se nakon što je aktivnost pauzirana, a i dalje je vidljiva, dok je fokusu nešto drugo (na primer: dijalog)
5	onStop() Zove se kada aktivnost više nije vidljiva
6	onDestroy() Zove se nakon što je aktivnost uništena od strane sistema
7	onRestart() Zove se nakon što se aktivnost restartuje nakon stopiranja

Primer - životni ciklus aplikacije

MainActivity.kt - override-ovati metode od značaja

Pratiti sadržaj kozole u sledećim situacijama

- Pri pokretanju aplikacije
- Nakon stavljanja aplikacije u background
- Nakon povratka aplikacije iz background-a
- Nakon klika na back

Dodavanje slike

```
@Composable
fun GreetingText(name: String, from: String, modifier: Modifier = Modifier) {
    Column(verticalArrangement = Arrangement.Center,
        modifier = modifier.fillMaxSize()) {
        Text(
            text = "Happy birthday $name!",
            fontSize = 20.sp,
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth()
        )
        Text(
            text = "From $from",
            fontSize = 16.sp,
            modifier = Modifier.align(alignment = Alignment.CenterHorizontally)
        )
    }
}
```

```
@Composable
fun Greeting(name: String, from: String, modifier: Modifier = Modifier){
    Column(verticalArrangement = Arrangement.Center,
        modifier = modifier.fillMaxSize()) {
        val image = painterResource(id = R.drawable.birthday_card)
        Image(painter = image, contentDescription = null,
            modifier = Modifier.align(alignment = Alignment.CenterHorizontally))
        GreetingText(name, from, modifier)
    }
}
```

